C 13

```
RRRRRRRRRRR    MMM        MMM   SSSSSSSSSSSS
RRRRRRRRRRR    MMM        MMM   SSSSSSSSSSSS
RRRRRRRRRRR    MMM        MMM   SSSSSSSSSSSS
RRR        RRR MMMMMM  MMMMMM SSS
RRR        RRR MMMMMM  MMMMMM SSS
RRR        RRR MMMMMM  MMMMMM SSS
RRR        RRR MMM   MMM  MMM SSS
RRR        RRR MMM   MMM  MMM SSS
RRR        RRR MMM   MMM  MMM SSS
RRRRRRRRRRR    MMM        MMM   SSSSSSSSS
RRRRRRRRRRR    MMM        MMM   SSSSSSSSS
RRRRRRRRRRR    MMM        MMM   SSSSSSSSS
RRR  RRR       MMM        MMM            SSS
RRR  RRR       MMM        MMM            SSS
RRR  RRR       MMM        MMM            SSS
RRR     RRR    MMM        MMM            SSS
RRR     RRR    MMM        MMM            SSS
RRR     RRR    MMM        MMM            SSS
RRR        RRR MMM        MMM SSSSSSSSSSSS
RRR        RRR MMM        MMM SSSSSSSSSSSS
RRR        RRR MMM        MMM SSSSSSSSSSSS
```

```
**FILE**ID**RM2EXTEND                                  K  2




RRRRRRRR    MM      MM    222222    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
RRRRRRRR    MM      MM    222222    EEEEEEEEEE  XX      XX  TTTTTTTTTT  EEEEEEEEEE  NN      NN  DDDDDDDD
RR      RR  MMMM  MMMM  22      22  EE          XX      XX      TT      EE          NN      NN  DD      DD
RR      RR  MMMM  MMMM  22      22  EE          XX      XX      TT      EE          NN      NN  DD      DD
RR      RR  MM  MM  MM          22  EE            XX  XX        TT      EE          NNNN    NN  DD      DD
RR      RR  MM  MM  MM          22  EE            XX  XX        TT      EE          NNNN    NN  DD      DD
RRRRRRRR    MM      MM        22    EEEEEEEE        XX          TT      EEEEEEEE    NN  NN  NN  DD      DD
RRRRRRRR    MM      MM        22    EEEEEEEE        XX          TT      EEEEEEEE    NN  NN  NN  DD      DD
RR  RR      MM      MM      22      EE            XX  XX        TT      EE          NN    NNNN  DD      DD
RR  RR      MM      MM      22      EE            XX  XX        TT      EE          NN    NNNN  DD      DD
RR    RR    MM      MM    22        EE          XX      XX      TT      EE          NN      NN  DD      DD      ....
RR    RR    MM      MM    22        EE          XX      XX      TT      EE          NN      NN  DD      DD      ....
RR      RR  MM      MM  2222222222  EEEEEEEEEE  XX      XX      TT      EEEEEEEEEE  NN      NN  DDDDDDDD        ....
RR      RR  MM      MM  2222222222  EEEEEEEEEE  XX      XX      TT      EEEEEEEEEE  NN      NN  DDDDDDDD        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
    (2)         92        DECLARATIONS
    (3)        120        RM$LOCK_PROLOG - ROUTINE TO LOCK PROLOG AND CHECK FOR EXTEND O.K.
    (6)        220        RM$UPD_PROLOG2 - ROUTINE TO UPDATE AND REWRITE PROLOG
```

```
0000        $BEGIN  RM2EXTEND,000,RM$RMS2,<PROLOG LOCK AND UPDATE ROUTINES>
0000
0000  ;
0000  ;****************************************************************************
0000  ;*                                                                          *
0000  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000  ;*  ALL RIGHTS RESERVED.                                                    *
0000  ;*                                                                          *
0000  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000  ;*  TRANSFERRED.                                                            *
0000  ;*                                                                          *
0000  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000  ;*  CORPORATION.                                                            *
0000  ;*                                                                          *
0000  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000  ;*                                                                          *
0000  ;*                                                                          *
0000  ;****************************************************************************
0000  ;
0000  ;++
0000  ; Facility: RMS32
0000  ;
0000  ; Abstract:
0000  ;
0000  ;       This module provides two routines to handle the locking
0000  ;       and updating of file prologs on extends.
0000  ;
0000  ; Environment:
0000  ;               STAR PROCESSOR RUNNING STARLET EXEC.
0000  ;
0000  ;
0000  ; Author: L F Laverdure,          Creation date: 1-DEC-1977
0000  ;
0000  ; Modified By:
0000  ;
0000  ;       V03-006 JEJ0050         J E Johnson             24-Jul-1984
0000  ;               Fix broken error paths in RM$LOCK_PROLOG to keep from
0000  ;               releasing the same BDB/BLB twice on an error exit.
0000  ;
0000  ;       V03-005 RAS0284         Ron Schaefer            30-Mar-1984
0000  ;               Fix STV value on error paths for RMS$_RPL and RMS$_WPL errors.
0000  ;
0000  ;       V03-004 KBT0333         Keith B. Thompson       10-Sep-1982
0000  ;               Remove $FRBDEF
0000  ;
0000  ;       V03-003 KBT0220         Keith B. Thompson       23-Aug-1982
0000  ;               Reorganize psects
0000  ;
0000  ;       V03-002 KBT0118         Keith B. Thompson       6-Aug-1982
0000  ;               Remove ref. to set_sifb_adr
```

```
0000   ;
0000   ;       V03-001 KBT0092         Keith B. Thompson       13-Jul-1982
0000   ;               Clean up psects
0000   ;
0000   ;       V02-010 CDS0003         C Saether               10-Dec-1981
0000   ;               Fix broken branch.
0000   ;
0000   ;       V02-009 CDS0002         C. Saether              28-Aug-1981
0000   ;               Modifications for new cache/release routines.
0000   ;               Use RM$RLNER1 entry point now.
0000   ;
0000   ;       V02-008 CDS0001         C. Saether              17-Apr-1981
0000   ;               Fix error path on prologue write errors to not release
0000   ;               bucket twice.
0000   ;
0000   ;       V02-007 REFORMAT        Maria del C. Nasr       2-Aug-1980
0000   ;
0000   ;       V006    CDS0049         C Saether               31-OCT-1979     19:45
0000   ;               Lock BDB allocated at connect time now.  Remove code to
0000   ;               allocate and deallocate on the fly.
0000   ;
0000   ;       V005    CDS0025         C Saether               27-jul-1979     5:30
0000   ;               interlock on vbn -1 if sharing sequential file
0000   ;
0000   ;       V004    CDS0001         C D Saether             16-Mar-1979     16:10
0000   ;               get bcb on extend when shared, don't release prologue
0000   ;               twice on error
0000   ;
0000   ;       V003    RAN0003         R A Newell              9-Nov-1978      10:36
0000   ;               file sharing code enhancements
0000   ;--
0000   ;
0000
```

```
0000          .SBTTL  DECLARATIONS
0000
0000  ;
0000  ; Include Files:
0000  ;
0000
0000  ;
0000  ; Macros:
0000  ;
0000
0000          $BDBDEF
0000          $CSHDEF
0000          $FABDEF
0000          $IFBDEF
0000          $IRBDEF
0000          $PLGDEF
0000          $RLSDEF
0000          $RMSDEF
0000
0000  ;
0000  ; Equated Symbols:
0000  ;
0000
0000  ;
0000  ; Own Storage:
0000  ;
0000
```

```
0000          .SBTTL  RM$LOCK_PROLOG - ROUTINE TO LOCK PROLOG AND CHECK FOR EXTEND O.K.
0000
0000    ;++
0000    ; RM$LOCK_PROLOG
0000    ;
0000    ;  This routine causes the prolog (vbn 1) for a relative file
0000    ;  to be locked for an extend operation.
0000    ;  The prolog is read in, its checksum is verified, the noextend flag is checked
0000    ;  for being clear if this is a relative file, a lock bdb is allocated, and
0000    ;  the buffer for the prolog is released keeping vbn 1 locked.
0000    ;
0000    ; Calling sequence:
0000    ;
0000    ;       bsbw    rm$lock_prolog
0000    ;
0000    ; Input Parameters:
0000    ;
0000    ;       r11     impure area address
0000    ;       r10     ifab address
0000    ;       r9      ifab/irab address
0000    ;       r8      fab/rab address
0000    ;
0000    ; Implicit Inputs:
0000    ;
0000    ;       none
0000    ;
0000    ; Output Parameters:
0000    ;
0000    ;       r0                      status code
0000    ;       r4                      lock bdb address
0000    ;       r1-r3,r5,ap             destroyed
0000    ;
0000    ; Implicit Outputs:
0000    ;
0000    ;       none
0000    ;
0000    ; Completion Codes:
0000    ;
0000    ;       standard rms.
0000    ;
0000    ; Side Effects:
0000    ;
0000    ;       none
0000    ;
0000    ;--
0000
```

```
                                 0000
                                 0000   ;
                                 0000   ;  This is not an entry point to this module.
                                 0000   ;
                                 0000
                                 0000   SEQLCK:
             51     01     CE    0000            MNEGL    #1,R1                       ; -1 to r1
                    52     D4    0003            CLRL     R2                          ; no size
                                 0005            $CACHE   VBN=R1,SIZE=R2,-
                                 0005                      FLAGS=<LOCK,NOREAD,NOBUFFER>
                                 000B                                                 ; lock on vbn -1
                           05    000B            RSB                                  ; and return
                                 000C
                                 000C   RM$LOCK_PROLOG::
             6A     38     E0    000C            BBS      #IFB$V_SEQFIL,(R10),-
                    F0           000F                      SEQLCK                     ; branch if sequential file
                                 0010            $CACHE   VBN=#1,SIZE=#512,-
                                 0010                      FLAGS=LOCK,ERR=ERRRPE      ; read & lock prolog
                FFDC'      30    0021            BSBW     RM$CHKSUM                   ; verify checksum
                28 50      E9    0024            BLBC     R0,ERROR
                                 0027            ASSUME   PLG$V_NOEXTEND EQ 0
          07 10 A5         E9    0027            BLBC     PLG$B_FLAGS(R5),10$         ; extends allowed
                                 002B            RMSERR   EXT                         ; extends not allowed
                    1D     11    0030            BRB      ERROR                       ; and out
                                 0032   10$:
             53     04     D0    0032            MOVL     #RLS$M_KEEP_LOCK,R3         ; specify flag for release
                  FFC8'    30    0035            BSBW     RM$RELEASE                  ; release buffer for vbn 1
                           05    0038            RSB
                                 0039
                                 0039   ;
                                 0039   ;  error reading prolog  -  change error code
                                 0039   ;
                                 0039
                                 0039   ERRRPE:
                 0C A8     D5    0039            TSTL     FAB$L_STV(R8)               ; do we have an stv?
                    09     12    003C            BNEQ     10$                         ; okay use it
  0C A8    50    00001000 8F  C9    003E            BISL3    #^X1000,R0,FAB$L_STV(R8); else set the RMS error there
                                 0047   10$:     RMSERR   RPL
                    54     D4    004C            CLRL     R4                          ; don't release bdb/blb again in put.
                           05    004E            RSB
                                 004F
                                 004F   ;
                                 004F   ;  handle checksum error by releasing lock on prolog
                                 004F   ;
                                 004F
                                 004F   ERROR:
                    50     DD    004F            PUSHL    R0                          ; save status code
                  FFAC'    30    0051            BSBW     RM$RLNER1                   ; release lock
                  50 8ED0        0054            POPL     R0                          ; restore status
                    54     D4    0057            CLRL     R4                          ; don't release again in put
                           05    0059            RSB                                  ; return to caller
                                 005A
```

```
005A            .SBTTL  RM$UPD_PROLOG2 - ROUTINE TO UPDATE AND REWRITE PROLOG
005A
005A    ;++
005A    ;
005A    ; RM$UPD_PROLOG2
005A    ; RM$SETHEBK
005A    ; RM$RLSPLG
005A    ;
005A    ;  This routine causes the prolog (vbn 1) for a relative file to be updated
005A    ;  and rewritten to the file, releasing the lock on the prolog.
005A    ;  If the status code on entry to this routine indicates an error, the
005A    ;  noextend bit is set in the prolog indicating that the file cannot be extended.
005A    ;
005A    ; Calling sequence:
005A    ;
005A    ;       bsbw    rm$upd_prolog2
005A    ;
005A    ;  alternate entry at rm$rlsplg to merely release the prolog lock.
005A    ;
005A    ; Input Parameters:
005A    ;
005A    ;       r11     impure area address
005A    ;       r10     ifab address
005A    ;       r9      ifab/irab address
005A    ;       r8      fab/rab address
005A    ;       r6      new eof vbn (= high block + 1)
005A    ;       r0      status code
005A    ;
005A    ; Implicit Inputs:
005A    ;
005A    ;       It is assumed that the prolog is already locked.
005A    ;
005A    ; Output Parameters:
005A    ;
005A    ;       r0              status code
005A    ;       r1-r5,ap        destroyed
005A    ;
005A    ; Implicit Outputs:
005A    ;
005A    ;       IFB$L_HBK and IFB$L_EBK are set from (R6 - 1) and R6 respectively.
005A    ;       For sequential files being shared, IFB$L_EBK is updated from
005A    ;       IRB$L_CURVBN.
005A    ;       The lock BDB is deallocated.
005A    ;
005A    ; Completion Codes:
005A    ;
005A    ;       standard rms.
005A    ;
005A    ; Side Effects:
005A    ;
005A    ;       Any stream waiting on the prolog will be restarted.
005A    ;
005A    ;--
005A
```

```
                              005A    275 RM$UPD_PROLOG2::
                   50    DD   005A    276             PUSHL    R0                              ; save status
                              005C    277             $CACHE   VBN=#1,SIZE=#512,-
                              005C    278                       FLAGS=LOCK,ERR=ERRRPL         ; reread prolog blk 1
                 FF90'   30   006D    279             BSBW     RM$CHKSUM                       ; verify checksum
                2E 50    E9   0070    280             BLBC     R0,ERRRPL                       ; branch if bad
                4C 6E    E9   0073    281             BLBC     (SP),ERREXT                     ; branch if error on entry
           70 A5   56    D0   0076    282             MOVL     R6,PLG$L_EOF(R5)                ; set eof vbn
                 FF83'   30   007A    283 MAKSUM: BSBW     RM$MAKSUM                       ; recompute checksum
                   03    88   007D    284             BISB2    #BDB$M_DRT!BDB$M_VAL,-
                0A A4         007F    285                       BDB$B_FLGS(R4)                ; say buffer valid & dirty
              53   06    D0   0081    286             MOVL     #RLS$M_WRT_THRU!-
                              0084    287                       RLS$M_KEEP_LOCK,R3            ; cause immediate write
                 FF79'   30   0084    288             BSBW     RM$RELEASE                      ; write vbn 1
                47 50    E9   0087    289             BLBC     R0,ERRWPL                       ; branch if error
                              008A    290
                              008A    291 ;
                              008A    292 ;  update eof vbn and highest allocated block
                              008A    293 ;
                              008A    294
                              008A    295 RM$SETHEBK::
       70 AA   56   01   C3   008A    296             SUBL3    #1,R6,IFB$L_HBK(R10)                    ; update hi block
           0A 6A   38    E0   008F    297             BBS      #IFB$V_SEQFIL,(R10),SEQFL               ; branch if really seq
           74 AA   56    D0   0093    298             MOVL     R6,IFB$L_EBK(R10)                       ; update eof block
                              0097    299
                              0097    300 ;
                              0097    301 ;  release lock on prolog
                              0097    302 ;
                              0097    303
                 FF66'   30   0097    304 RLSPLG: BSBW     RM$RLNER1                               ; free lock on prolog
                   01    BA   009A    305             POPR     #^M<R0>                                 ; restore status code
                         05   009C    306             RSB                                              ; return to caller
                              009D    307 SEQFL:
                   5A    DD   009D    308             PUSHL    R10                                     ; save ifab addr for what i dont know
                   F6    11   009F    309             BRB      RLSPLG                                  ; go release vbn -1
```

```
RM2
Sym

$$.
$$.
$$R
$$R
$$R
$$R
BDB
BDB
BDB
CSH
CSH
CSH
ERR
ERR
IFB
RLS
RM$
RM$
RM$
RM$


PSE
---
.
RM$
$AB


Pha
---
Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
245
The
203
15
```

```
                                 00A1
                                 00A1    ;
                                 00A1    ;  Handle prolog read error
                                 00A1    ;  Prolog may still be locked.  Must release it.
                                 00A1    ;
                                 00A1
                                 00A1    ERRRPL:
                                 00A1            $CACHE  VBN=#1,SIZE=#0,-
                                 00A1                     FLAGS=NOREAD                   ; find prolog bdb
                    E8 6E    E9  00AC            BLBC    (SP),RLSPLG                     ; branch if already had error
                    0C A8    D5  00AF            TSTL    FAB$L_STV(R8)                   ; do we have an stv?
                       09    12  00B2            BNEQ    10$                             ; okay use it
 0C A8    6E  00001000 8F    C9  00B4            BISL3   #^X1000,(SP),FAB$L_STV(R8); else set the RMS error there
                                 00BD    10$:    RMSERR  RPL,(SP)                        ; replace status code
                                 00C2
                                 00C2    ;
                                 00C2    ;  Caller entered rm$upd_prolog2 with an error status.
                                 00C2    ;  This must have been due to a bucket format write failure.
                                 00C2    ;  Set the noextend bit in the prolog to prevent further attempts
                                 00C2    ;  to extend the file.
                                 00C2    ;
                                 00C2
                                 00C2    ERREXT:
                       01    88  00C2            BISB2   #1@PLG$V_NOEXTEND,-
                    10 A5        00C4             PLG$B_FLAGS(R5)                ; set flag
                                 00C6            RMSERR  EXT,(SP)                        ; update error code
              56    74 AA    D0  00CB            MOVL    IFB$L_EBK(R10),R6               ; cause ebk not to be updated
                       A9    11  00CF            BRB     MAKSUM                          ; go rewrite prolog
                                 00D1
                                 00D1    ;
                                 00D1    ;  Error occurred trying to rewrite the prolog.
                                 00D1    ;  Update status code and return.  The bdb was already
                                 00D1    ;  released in release due to the error.
                                 00D1    ;
                                 00D1
                                 00D1    ERRWPL:
                    13 6E    E9  00D1            BLBC    (SP),20$                        ; branch if already had error
                    0C A8    D5  00D4            TSTL    FAB$L_STV(R8)                   ; do we have an stv?
                       09    12  00D7            BNEQ    10$                             ; okay use it
 0C A8    6E  00001000 8F    C9  00D9            BISL3   #^X1000,(SP),FAB$L_STV(R8); else set the RMS error there
                                 00E2    10$:    RMSERR  WPL,(SP)                        ; change status code
                       01    BA  00E7    20$:    POPR    #^M<R0>                         ; pop status off stack
                             05  00E9            RSB                                     ; return
                                 00EA
                                 00EA    ;++
                                 00EA    ;  Entry point to release the prolog.
                                 00EA    ;
                                 00EA    ;  inputs:
                                 00EA    ;       r4      bdb address
                                 00EA    ;       r0      status code
                                 00EA    ;
                                 00EA    ;  outputs:
                                 00EA    ;       R1-R5, AP destroyed
                                 00EA    ;
                                 00EA    ;--
                                 00EA
                                 00EA    RM$RLSPLG::
```

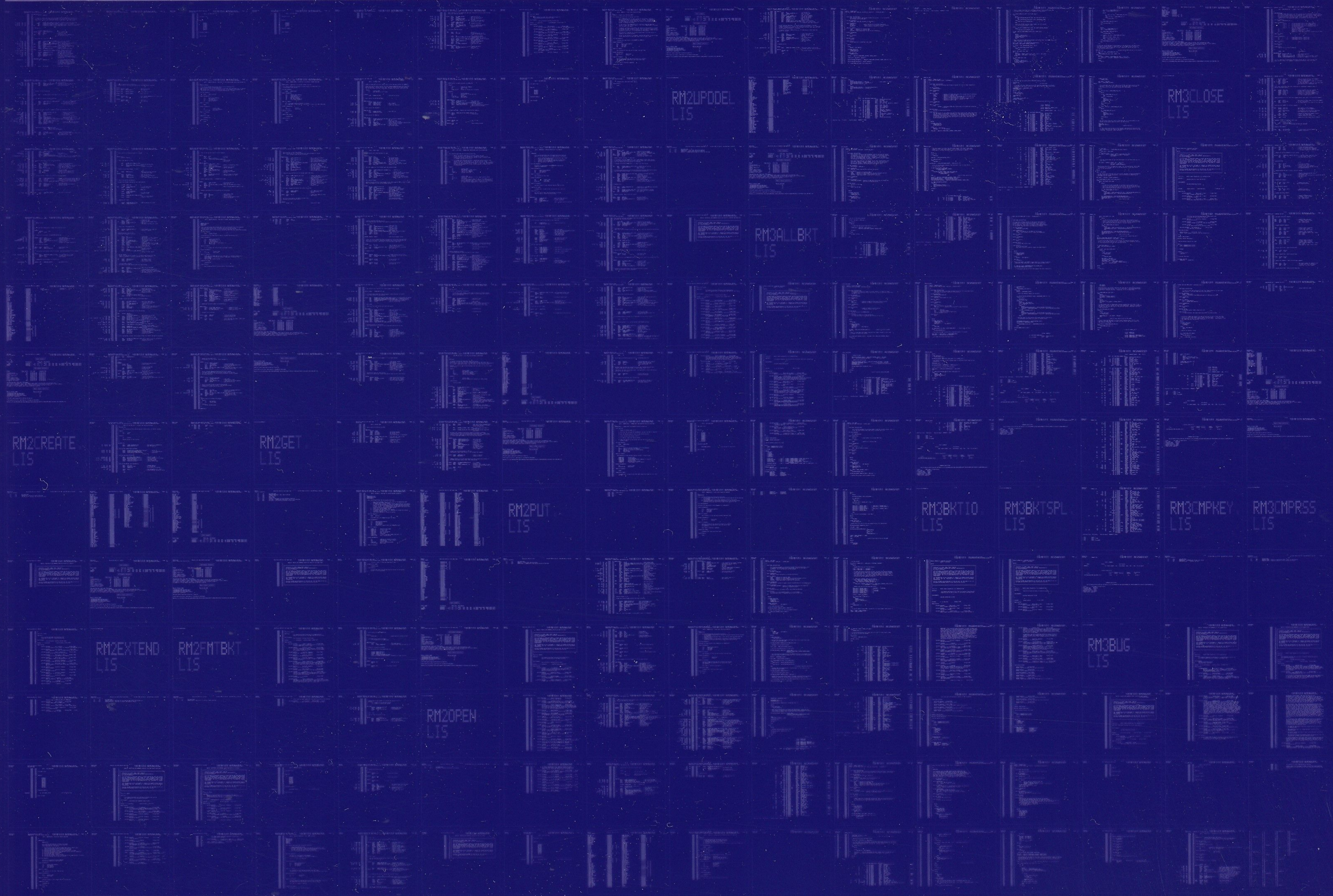
```
                       50   DD  00EA    368         PUSHL   R0                           ; push status
                       A9   11  00EC    369         BRB     RLSPLG                       ; go do release
                                00EE    370
                                00EE    371
                                00EE    372         .END
```

```
$$.PSECT_EP                      = 00000000
$$.TMP                           = 00000004
$$RMSTEST                        = 0000001A
$$RMS_PBUGCHK                    = 00000010
$$RMS_TBUGCHK                    = 00000008
$$RMS_UMODE                      = 00000004
BDB$B_FLGS                       = 0000000A
BDB$M_DRT                        = 00000002
BDB$M_VAL                        = 00000001
CSH$M_LOCK                       = 00000001
CSH$M_NOBUFFER                   = 00000008
CSH$M_NOREAD                     = 00000004
ERREXT                             000000C2 R        01
ERROR                              0000004F R        01
ERRRPE                             00000039 R        01
ERRRPL                             000000A1 R        01
ERRWPL                             000000D1 R        01
FAB$L_STV                        = 0000000C
IFB$L_EBK                        = 00000074
IFB$L_HBK                        = 00000070
IFB$V_SEQFIL                     = 00000038
MAKSUM                             0000007A R        01
PLG$B_FLAGS                      = 00000010
PLG$L_EOF                        = 00000070
PLG$V_NOEXTEND                   = 00000000
RLS$M_KEEP_LOCK                  = 00000004
RLS$M_WRT_THRU                   = 00000002
RLSPLG                             00000097 R        01
RM$CACHE                           ********    X     01
RM$CHKSUM                          ********    X     01
RM$LOCK_PROLOG                     0000000C RG       01
RM$MAKSUM                          ********    X     01
RM$RELEASE                         ********    X     01
RM$RLNER1                          ********    X     01
RM$RLSPLG                          000000EA RG       01
RM$SETHEBK                         0000008A RG       01
RM$UPD_PROLOG2                     0000005A RG       01
RMS$_EXT                         = 0001C022
RMS$_RPL                         = 0001C104
RMS$_WPL                         = 0001C11C
SEQFL                              0000009D R        01
SEQLCK                             00000000 R        01

                                          +----------------+
                                          ! Psect synopsis !
                                          +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
RM$RMS2                          000000EE  (  238.)  01 (  1.)    PIC   USR   CON   REL   GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                            00000000  (    0.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD     WRT NOVEC BYTE
```

```
                                              +--------------------------+
                                              ! Performance indicators !
                                              +--------------------------+

Phase                        Page faults     CPU Time        Elapsed Time
-----                        -----------     --------        ------------
Initialization                      31       00:00:00.08     00:00:01.04
Command processing                 129       00:00:00.76     00:00:03.75
Pass 1                             278       00:00:08.68     00:00:24.97
Symbol table sort                    0       00:00:01.04     00:00:02.45
Pass 2                              74       00:00:01.72     00:00:04.01
Symbol table output                  6       00:00:00.08     00:00:00.41
Psect synopsis output                1       00:00:00.02     00:00:00.02
Cross-reference output               0       00:00:00.00     00:00:00.00
Assembler run totals               521       00:00:12.38     00:00:36.68

The working set limit was 1350 pages.
46784 bytes (92 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 865 non-local and 5 local symbols.
372 source lines were read in Pass 1, producing 13 object records in Pass 2.
22 pages of virtual memory were used to define 21 macros.

                                             +----------------------------+
                                             ! Macro library statistics !
                                             +----------------------------+

Macro library name                                   Macros defined
------------------                                   --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                                   13
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                                    0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                                 4
TOTALS (all libraries)                                           17

997 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:RM2EXTEND/OBJ=OBJ$:RM2EXTEND MSRC$:RM2EXTEND/UPDATE=(ENH$:RM2EXTEND)+EXECML$/LIB+LIB$:RMS/LIB
```

0323 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



RM2UPDDEL
LIS

RM3CLOSE
LIS

RM3ALLBKT
LIS

RM2CREATE
LIS

RM2GET
LIS

RM2PUT
LIS

RM3BKTIO
LIS

RM3BKTSPL
LIS

RM3CMPKEY
LIS

RM3CMPRSS
LIS

RM2EXTEND
LIS

RM2FMTBKT
LIS

RM3BUG
LIS

RM2OPEN
LIS